Pending Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently amended) A method of operating a processor comprising:

executing a branch instruction that causes a processor to branch from executing a first sequential series of instructions to a different sequential series of instructions based on a byte, specified by the branch instruction, in a register, specified by the branch instruction, being equal or not equal to a byte value[[,]] specified by the branch instruction~~, if the specified byte matches or mismatches the specified byte value~~.

2. (Previously presented) The method of claim 1 wherein the branch is to an instruction at a specified label.

3. (Previously presented) The method of claim 1 wherein the branch instruction comprises:

a bit_postion field that specifies the byte in a longword contained in the register.

4. (Previously presented) The method of claim 1 wherein the branch instruction comprises:

an optional token that is set by a programmer and specifies a number i of instructions to execute following the branch instruction before performing the branch operation.

5. (Previously presented) The method of claim 1 wherein the branch instruction comprises:

an optional token that is set by a programmer and specifies a number i of instructions to execute following the branch instruction before performing the branch operation where the number of instructions can be specified as one, two or three.

6. (Previously presented) The method of claim 1 wherein the register is a context-relative transfer register or a general-purpose register that holds the operand.

7. (Previously presented) The method of claim 1 wherein the branch instruction comprises:

an optional token that is set by a programmer and which specifies a guess_branch prefetch for the instruction for the "branch taken" condition rather than the next sequential instruction.

8. (Previously presented) The method of claim 1 wherein the branch instruction comprises:

an optional token that is set by a programmer and specifies a number i of instructions to execute following the branch instruction before performing the branch operation; and

a second optional token that is set by a programmer and which specifies a guess_branch prefetch for the instruction for the "branch taken" condition rather than the next sequential instruction.

9. (Previously presented) The method of claim 1 wherein the branch instruction allows a programmer to specify which bit of the register to use to determine the branch operation.

10. (Previously presented) The method of claim 1 wherein the branch instructions allows branches to occur based on evaluation of a byte that is in a data path of a processor.

11. (Previously presented) The method of claim 1 wherein the branch instruction branches on the byte matching the byte value and wherein the instruction prefetches the instruction for the "branch taken" condition.

12. (Previously presented) The method of claim 1 wherein the branch instruction branches on the byte not matching the byte value and wherein the instruction prefetches the next sequential instruction.

13. (Previously presented) The method of claim 1 wherein the branch instruction includes a Byte_spec Number that specifies the byte in the register to be compared with byte_compare_value.

14. (Previously presented) A computer program product residing on a computer readable medium comprising instructions, including a branch instruction that causes a processor to:

fetch a byte, specified by the branch instruction, stored in a register, specified by the branch instruction;

determine whether the byte in the register is equal or not equal to a specified byte value contained in the branch instruction; and

perform a branching operation specified by the branch instruction based on the specified byte being equal or not equal to the byte in the register.

15. (Original) The product of claim 14 wherein the branch is to an instruction at a specified label.

16. (Previously presented) The product of claim 14 wherein the branch instruction comprises:

a bit_postion field that specifies the byte in a longword contained in the register.

17. (Previously presented) A processor comprises:

a register stack;

an arithmetic logic unit coupled to the register stack and a program control store that stores a branch instruction that causes the processor to:

fetch a byte, specified by the branch instruction, stored in a register, specified by the branch instruction;

determine whether the byte in the register is equal or not equal to a specified byte value contained in the branch instruction; and perform a branching operation specified by the branch instruction based on the specified byte being equal or not equal to the byte in the register.

18. (Previously presented) The processor of claim 17 wherein the branch instruction that causes the processor to perform the branching operation causes the processor to branch to an instruction at a specified label.

19. (Previously presented) The processor of claim 17 wherein a bit_postion field in the branch instruction specifies the byte in a longword contained in the register.

20. (Previously presented) A method of operating a processor comprises:

executing a branch instruction by:

fetching a byte, specified by the branch instruction, stored in a register, specified by the branch instruction;

determining whether the byte in the register is equal or not equal to a specified byte value contained in the branch instruction; and

performing a branching operation specified by the branch instruction based on the specified byte being equal or not equal to the byte in the register.

21. (Previously presented) The method of claim 20 wherein performing the branch includes branching to an instruction at a specified label.